ELSEVIER

Microelectronics Journal **27** (1996) 11–22

0026-2692/96/$15.00

0026-2692(95)00008-9

Research paper

2D matrix multiplication on a 3D systolic array

Salim Lakhani[1], Yi Wang[1], Aleksander Milenković[2] and Veljko Milutinović[2]

[1] *School of Electrical Engineering, Purdue University, West Lafayette, Indiana, USA*
[2] *School of Electrical Engineering, University of Belgrade, Belgrade, Serbia*

The introduction of systolic arrays in the late 1970s had an enormous impact on the area of special purpose computing. However, most of the work so far has been done with one-dimensional and two-dimensional (2D) systolic arrays. Recent advances in three-dimensional VLSI (3D VLSI) and 3D packaging of 2D VLSI components, has made the idea of 3D systolic arrays feasible in the near future. In this paper we introduce one algorithm for 2D matrix multiplication, using a 3D systolic array. We analyze advantages and disadvantages of 3D systolic arrays in the context of the analysis algorithm. The analytical work is combined with examples and discussions of relevant details.

1. Introduction

Recent advances in VLSI technology have made it possible to use special purpose processors to solve compute-bound problems [1]. If a systolic array architecture is used, simple and regular processing elements (or cells) capable of doing simple computations are connected using a nearest-neighbor network. In these arrays, data pass through many cells, and are used by different cells for computation, before being returned to the memory [2]. As the same data are used repeatedly for many computations, the computational throughput is increased without a need for increasing the I/O bandwidth or using a local memory. Furthermore, since the cells of the systolic arrays are simple and regular, they are easier and cheaper to design.

Most work so far has been done with one-dimensional (1D) and two-dimensional (2D) systolic arrays, which we shall refer to as planar systolic arrays in the rest of this paper. Planar systolic arrays have been widely used in signal and image processing. However, there are some inherent limitations to the speed, extensibility and partitionability of planar systolic arrays.

One major problem with the planar systolic arrays is the speed limitation. In a planar systolic array, speed of the data stream is dictated by the speed of the computation in each cell. The speed of computation may be low, especially if the cells are required to do complex computations such as floating-point multiplication and/or addition. For this reason, when the size of the problem becomes large (and consequently the size of the array), the computational latency may become too large to be tolerable.

Another problem with the planar systolic arrays is difficulty with the extensibility. The structure of a systolic array is fixed after it has been manu-

factured, so it has less flexibility than SIMD machines. Since it is impossible to produce an array to match all possible sizes of different problems, it is necessary that one is able to solve a larger problem on a smaller array. To be able to solve a larger problem on a smaller array, a bus from the output to the input of the array may be needed to feed back the partial results. This is especially true if partial results and data streams flow in the same plane, which is generally true for a planar systolic array. This bus can not only affect the latency due to the propagation delay, but it can also increase the hardware cost.

Still another problem with the planar systolic array is difficulty with the partitionability. This problem arises when one needs to solve a problem of smaller size on an array of larger size. Using a larger array to solve a problem of smaller size is a waste of resources, therefore it is important that one is able to solve several smaller size problems on a larger array simultaneously. This may be difficult if partial results and data streams flow in the same plane, which is generally true for a planar systolic array.

To solve these inherent limitations of the planar systolic arrays, researchers have recently turned their attention to three-dimensional (3D) systolic arrays. This was made possible by recent advances in 3D VLSI and related areas. Important contributions in the technology domain, among others, came from IBM [3], Texas Instruments [4] and Hughes [5].

The pioneering work in the area of 3D computer engineering/science was done by Aboelaze, Kaczorek, Leighton, Preparata, Rosenberg and Wah [6–12]. Rosenberg has been working on the modeling of 3D layouts and layouts of selected interconnection networks, Preparata has also been working on the layouts of selected interconnections networks, and Leighton and Rosenberg, as well as Aboelaze and Wah, have been working on various aspects of layout complexities.

The 3D systolic array is a concept in computer architecture. A 3D systolic array can be implemented with 3D VLSI, but 3D VLSI is not the only way to implement the 3D systolic array. The 3D systolic arrays can also be implemented using 3D packaging of 2D VLSI.

In this paper we focus on 2D and 3D systolic arrays for 2D matrix multiplication. In Section 2 we analyze the advantages of 3D systolic arrays, in Section 3 we discuss some of the possible problems with 3D systolic arrays, in Section 4 we define criteria to compare 2D and 3D systolic arrays, and in Section 5 we show the structure of the 3D systolic array for multiplication of 2D matrices, and we compare that structure with the structure of the corresponding 2D systolic array.

2. Advantages

The 3D systolic arrays have many potential advantages over the planar systolic arrays. These advantages include, but are not limited to, higher speed, better extensibility, better partitionability, better fault tolerance capabilities, ease of pipelining, ease of cascading, etc. Also the processing element for 3D systolic arrays may be simpler than that for planar systolic arrays. Some of the advantages are due to the unique 3D architecture of 3D systolic arrays, and the others are due to 3D VLSI and 3D packaging technologies. Even though some of the advantages of 3D systolic arrays are due to 3D VLSI and 3D packaging, most of the advantages of the 3D systolic array are due to its unique architectural concept.

In a 3D systolic array, constant (or direct) data streams move in the X–Y plane, and variable (or functional) data streams move along the Z axis (Fig. 1). All computations in a 3D systolic array are done in the cells along the Z axis [13]. As only constant data flow in the X–Y plane, they can move at higher speed, resulting in a reduction of the propagation delay. Also, due to the fact that computation in a cell depends only on

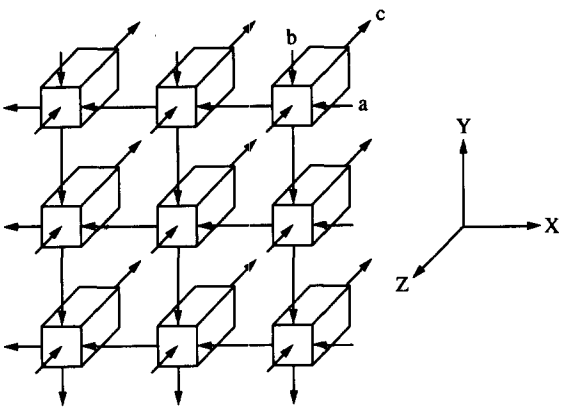

Fig. 1. A 3D systolic array. Note that constant data streams (a) and (b) are flowing in the X–Y plane and the varying data stream (c) is flowing along the Z axis. Also note that I/O can occur at any of the six planar sides of the array [13].

the constant data streams moving in the X–Y plane, computation can be done concurrently, therefore speed of the 3D systolic arrays will increase compared with the 2D systolic arrays.

As already indicated, some of the advantages of the 3D systolic array are due to the use of 3D VLSI and/or 3D packaging technologies. With 3D VLSI and/or 3D packaging technologies, components can be placed at a shorter distance from each other, which results in shorter wires between the components (Fig. 2) [14] and in smaller propagation delays. The 3D VLSI also has greater packing density than the 2D VLSI (Fig. 3), which means more components can be placed on the same chip [14]. More components on the same chip results in less need for off-chip communication, and thus higher throughput can be achieved. In 3D VLSI, we can also use holes instead of wires to connect components on different planes (Fig. 4 [14]), and hence we can connect more components without using long wires, which results in lower propagation delays. Note, in this paper the term layer is used for different VLSI mask levels, and the term plane is used for different levels of processing elements. With 3D technologies we can also construct multiport devices [14]. Multiport devices have a

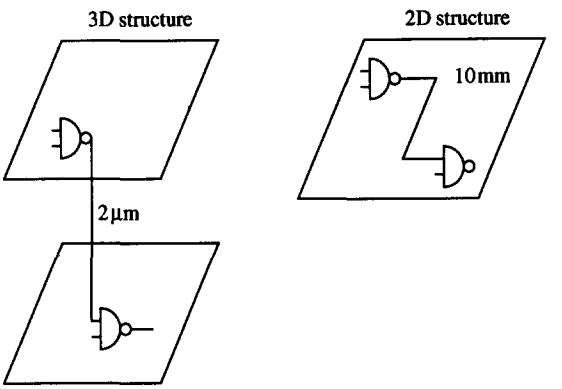

Fig. 2. Comparison of wiring lengths in a 2D structure and a 3D structure. Shorter wires are required in a 3D structure, as components can be placed close to each other by using more than one plane [14].

higher I/O bandwidth which increases the overall speed of the array.

Another advantage of a 3D systolic array is that it can be easily extended or partitioned. This ease of extensibility and partitionability is due to the fact that no partial results flow from one cell to the next in the X–Y plane [13]. As the computation depends only on the constant data flowing

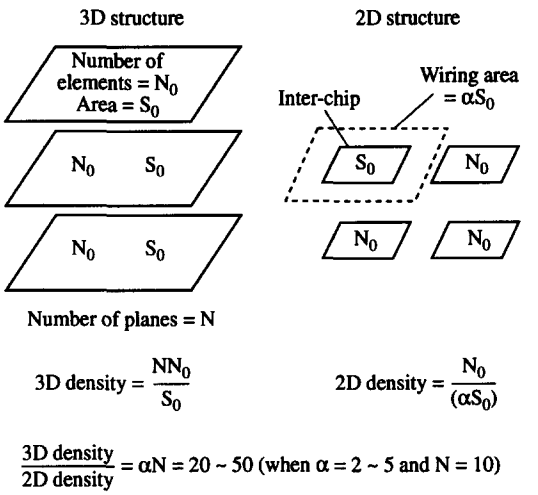

Fig. 3. Comparison of packing density of a 2D structure and a 3D structure. Note, the packing density of a 3D structure increases by a factor α, compared with a 2D structure [14].

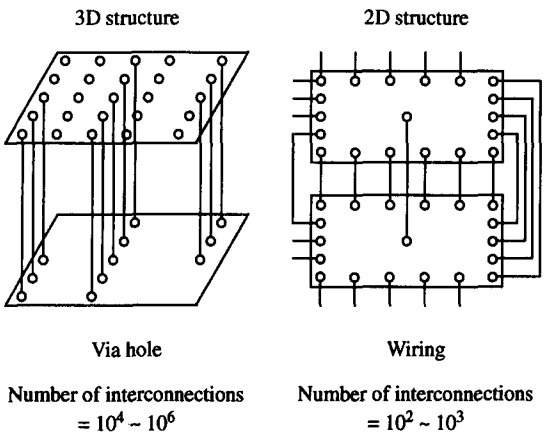

Fig. 4. Comparison of the possible number of interconnections in a 2D structure and a 3D structure. More components can be connected in a 3D structure as holes can be used, instead of wires, to connect components on different planes [14].

in the X–Y plane, an array can be extended or partitioned without affecting the result.

Sometimes it is necessary to combine two special purpose systolic arrays (in the form of a pipeline) to solve a complex problem. One example of such a pipeline is a systolic array to solve a system of linear equations. This macropipeline can be formed by combining a systolic array for LU decomposition of a matrix and a systolic array for solving a triangular linear system [2]. This can be easily accomplished with the 3D systolic arrays, because direct and functional data streams flow in different planes.

The next advantage is related to fault tolerance. One problem with single chip design is that, even if one small part of the chip develops a fault, the whole chip becomes useless. One solution is to use row reconfigurability (RR) and row–column reconfigurability (RCR) techniques. The RR and RCR techniques were proposed by Fortes and Raghavendra [15] for increasing the fault tolerance capabilities of a processor array. In this technique, if a processor develops a fault, it is bypassed by RR or RCR. This technique can be easily extended to 3D systolic arrays, and would work even better in 3D architectures than in planar architectures. In 3D systolic arrays, we can extend this technique to a full layer of processors. Thus, if a processor develops a fault we have three choices (as compared with two choices for a 2D systolic array and one choice for a 1D systolic array). For a 3D systolic array we can bypass a faulty processor in one of the following ways: (1) by row reconfiguration; (2) by column reconfiguration; or (3) by layer reconfiguration.

Here we deal only with time, area and area–time complexity-related issues. Details related to partitionability, extensibility and fault tolerance can be found in [16].

3. Disadvantages

The 3D systolic arrays also have some disadvantages. In this section we discuss some of the disadvantages of the 3D systolic arrays. We also present some of the possible ways to overcome these disadvantages. Some of these disadvantages are due to the architecture of the 3D systolic array and some are due to 3D VLSI. Three of the main disadvantages of the 3D systolic array are I/O problems, high cost (low yield) and heat dissipation.

The first problem of the 3D systolic array is I/O. This problem is partially due to the architecture of the 3D systolic array and partially due to 3D VLSI. For a planar systolic array, I/O can occur only at the four edges of the array, while for a 3D array I/O can occur at all six planar sides of the array (Fig. 1). This means that additional pins would be required for I/O, which is difficult with current packaging technologies. One solution for this problem is to use one pin for more than one purpose. This sharing of hardware may result in a reduction of the throughput for 3D arrays, but this reduced throughput may still be higher than that for 2D systolic arrays [13]. Another solution to this problem is to use 3D

VLSI and 3D packaging technologies, and at the same time advance 3D VLSI and 3D packaging technologies, so as to be able to have pins located on all four edges of more than one plane. This increase in the number of pins not only solves the I/O problem of the 3D arrays, but it can also result in an increase of I/O bandwidth, which in turn may improve the performance of the systolic array; but this increased I/O bandwidth of the 3D systolic array must be matched by the I/O bandwidth of the host (memory or computer) and the data bandwidth of the bus. If I/O bandwidth of the host or the data bandwidth of the bus is less than the I/O bandwidth of the 3D systolic array, then the host or the bus may become a bottleneck, and will result in degradation of the performance of the 3D systolic array. If the host is a memory, then one way to increase its I/O bandwidth is by using multiport memory. Bandwidth of a host memory can also be increased by using interleaved memory, memory pipelining, and other techniques. If the host is a computer, then the I/O bandwidth of the host and the 3D systolic array can be matched by using a buffer between the host computer and the 3D systolic array. Increasing the data bandwidth of the bus may be difficult. The only way to increase the data bandwidth of a bus is by using more than one bus between the host and the 3D systolic array. This use of more than one bus will result in an increase of the cost, therefore it is important to achieve a balance between speed of the 3D systolic array and its cost.

The second major problem of the 3D systolic array is the cost of the chip. This is due to 3D technology. In 3D technologies, the yield of the chip goes down as the number of planes increases [14]. This decrease in the yield results in an increase of the manufacturing cost. This problem can be avoided if one uses 2D VLSI instead of 3D VLSI. The use of 2D VLSI for the 3D systolic array may result in a decrease in the throughput, but this reduced throughput will be higher than the throughput of a planar systolic array. Therefore, due to the fact that manufacturing cost increases with increase in the number of planes in the 3D systolic array, it is important to achieve a balance between the utilization of 3D technologies and the cost of 3D systolic arrays.

The third major problem of the 3D systolic arrays, which is also due to 3D technology, is the problem of heat dissipation. This is due to the fact that it is difficult to dissipate the heat from the inner portions of a 3D VLSI chip. One solution to this problem is to restrict the number of planes in the third dimension. This problem can be avoided altogether if the 3D systolic arrays are implemented with 2D VLSI.

4. Comparison criteria

In this section we define criteria for comparing a planar and a 3D systolic array, to be used in the later analysis, and with respect to the previously mentioned advantages of the 3D approach. One simple but incorrect way to compare a 2D systolic array with a 3D systolic array would be by counting the number of cells in each. The main problem with this is that the cells in different types of systolic array may be different, therefore one can not draw any valid conclusions about the relative speed or hardware complexity of the two arrays just by counting the number of cells in the two arrays. Due to the differences in the structure of the cells, it is difficult to compare the planar and the 3D systolic arrays. For these reasons, it is important to develop some realistic and preferably analytical criteria that can be used for comparing the systolic arrays.

The two most common criteria for comparing two systolic arrays are processing latency (time T) and VLSI area (area A). We will use time and area to compare the systolic arrays. One must realize that usually there is a tradeoff involved between T and A, i.e. one hardware may take less time to solve a problem, but it may need more VLSI area than the slower hardware. Therefore, to understand clearly the differences between the systolic

arrays, one must also look at the overall performance. To compare this overall performance of the systolic arrays, we use the AT^2 criterion. Note, the AT^2 criterion is the adoption of the classical AT^2 criterion used in the theory of VLSI complexity analysis [17]. In this section, we present a general method for developing criteria to compare time and area. In Section 5, we use this general method to develop exact comparison criteria for our example.

The first step towards developing comparison criteria is to define precisely the algorithm for which one needs a systolic array. Then, for this algorithm, one has to define the molecular operation. Here, the molecular operation is defined as the operation that is constantly being repeated in the given algorithm, regardless of the type of systolic array being used. Next, one has to define atomic operations for all the systolic arrays that need to be compared. We define the atomic operations as the operations to be actually executed, by each cell of the systolic array, as elementary parts of the above-defined molecular operation. Note that the molecular and atomic operation for a particular array may or may not be the same. Now, one knows exactly what the operation to be performed by each cell of the systolic array is. Once one has this information, one can define the structure of the array and can develop the equations for time and area in terms of different relevant parameters. Next, one plots these equations versus the problem size, for different values of the chosen parameters. From these plots, one can analyze and compare the performance of the given systolic arrays.

4.1 Time

To develop an equation for the total execution time, one first has to define the time for one atomic cycle. Here atomic cycle is defined as the time needed by the slowest cell to perform its required atomic operation, and to pass the result/data to the next cell (unless all cells are characterized with the same delay, which is what the assumption is throughout the rest of this work). Note, the time for an atomic operation is the same as the time for an atomic cycle. Once one has defined the atomic cycle time, one can compute the time for the execution of the molecular operation, and for the execution of the whole algorithm. The total time is a function of the atomic cycle time and the problem size (plus a plethora of possible technology-related and algorithm-related parameters).

4.2 Area

To develop an equation for area, one first has to define the area for each cell of the systolic array. This area will be dictated by the complexity of the atomic operation. One already knows the number of cells in an array (from the structure of the array), thus one can now compute the total area of the systolic arrays. Note, in this work the area of a cell in a 3D systolic array is defined to be the area needed to implement the logic function of that cell in 2D VLSI. Thus, the area of a cell in a planar systolic array and the area of a cell in a 3D systolic array will be the same if both cells implement the same function. Therefore, here we actually talk about the virtual area, so that the planar and the 3D approaches can be compared, but (for simplicity) we use only the term area.

4.3 Area-time tradeoff

As we mentioned earlier, there is a tradeoff involved between area of the systolic array and the time for execution of the algorithm. Therefore, to understand the differences between the given systolic arrays, one must also look at the overall performance (in terms of area and time) of the given systolic arrays. To compare this overall performance of the given systolic arrays, we use the AT^2 criterion.

5. Matrix multiplication

In this section we present two different structures of 2D systolic arrays and one structure of a 3D systolic array for matrix multiplication. We

also compare the 2D and 3D systolic arrays for time and area according to the criteria defined in the previous section.

Given matrices **A** and **B** of size N by N, we need to compute matrix **C**, such that $\mathbf{C} = \mathbf{A} \cdot \mathbf{B}$. The algorithm for matrix multiplication is shown in Fig. 5. In this example, the molecular operation is $c + a \cdot b$. This operation will be performed by each cell of the 2D systolic array, therefore it is also the atomic operation for the 2D systolic array. Two different structures of the 2D systolic arrays (for matrix multiplication) are shown in Fig. 6a [18] and 6b [19]. The number of cells (n_{2a}) needed in the 2D systolic array of Fig. 6a, in terms of the problem size (N), is given by [13]:

$$n_{2a} = 3N^2 - 3N + 1$$

and the number of cells (n_{2b}) needed in the 2D systolic array of Fig. 6b, in terms of the problem size (N), is given by:

$$n_{2b} = N^2$$

One can decompose the molecular operation into atomic operations for a 3D systolic array. For example, assume that the elements of the matrices are floating-point numbers. Then one can decompose the molecular operation into five atomic operations. These atomic operations are: (1) mantissa multiplication; (2) exponent addition; (3) mantissa alignment; (4) mantissa addition; and (5) result normalization [13]. Mantissa multiplication and exponent addition are needed for the multiplication part of the molecular operation, and other atomic operations are needed for the addition part of the molecular

```
for (i=1; i≤N; i++) {
   for (j=1; j≤N; j++) {
      for (k=1; k≤N; k++) {
         c_{i,j} = c_{i,j} + a_{i,k} * b_{k,j};
         }
      }
   }
```

Fig. 5. An algorithm for matrix multiplication. Here $a_{i,j}$, $b_{i,j}$ and $c_{i,j}$ are the (i,j)th elements of matrices **A**, **B** and **C**, respectively.

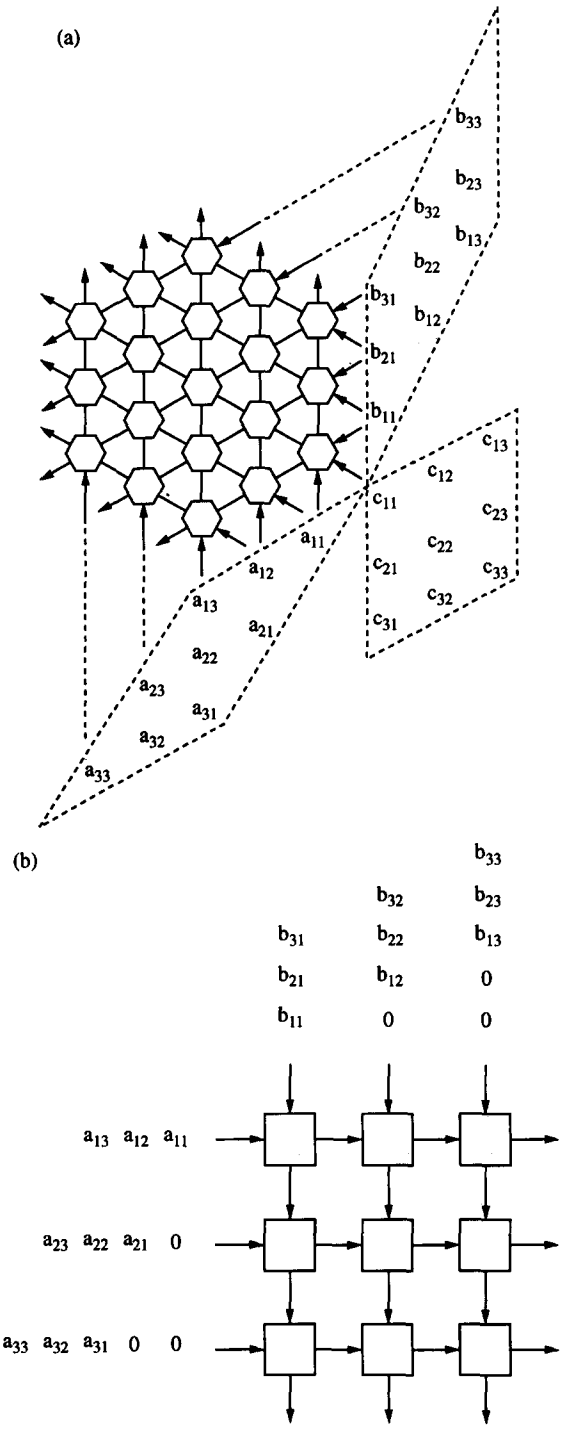

Fig. 6. Two different 2D systolic arrays for multiplying two 3 × 3 matrices: (a) [18]; (b) [19].

operation. Note, one can also use a molecular operation as an atomic operation, in which case all cells of the 3D systolic array will do the same operation, and the resulting array will be the same as the 2D systolic array of Fig. 6b. A 3D systolic array for matrix multiplication is shown in Fig. 7. Note, in the array of Fig. 7, there are five planes. The cells in the first plane perform the mantissa multiplication, the cells in the second plane perform the exponent addition, the cells in the third plane perform the mantissa alignment, the cells in the fourth plane do the mantissa addition, and the cells in the fifth plane normalize the result. For the 3D systolic array, the number of cells (n_3) is given by [13]:

$$n_3 = MN^2$$

where N is the problem size and M is the number of different cell planes (or the number of atomic operations, if cells in one plane execute only one atomic operation).

Now that we have defined the structure of both the 2D and the 3D systolic arrays (and we already have other necessary information), we

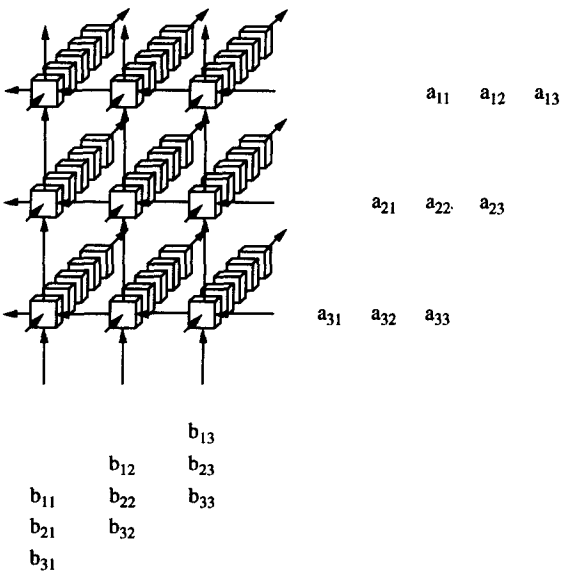

Fig. 7. A 3D systolic array for multiplying two 3 × 3 matrices.

can develop equations for time and area, to compare 2D and 3D systolic arrays.

5.1 Comparison of time

First, assume that all atomic operations are of the same length and the time needed for an atomic operation in the 2D systolic array is t_2. Then the atomic cycle for the 2D systolic array will be t_2. Note, as the molecular operation is the same as the atomic operation for the 2D systolic array, the time needed for molecular operation is also t_2. Now the total time (T_{2a}) needed for computation (of the matrix **C**) with the 2D systolic array of Fig. 6a will be equal to [13]:

$$T_{2a} = (3N - 2)t_2$$

and the total time (T_{2b}) needed for computation (of the matrix **C**) with the 2D systolic array of Fig. 6b will be equal to:

$$T_{2b} = (3N - 2)t_2$$

Note, the latency for both structures of 2D systolic arrays for matrix multiplication is the same. Therefore, in the rest of this section, we shall refer to both T_{2a} and T_{2b} simply as T_2.

For the 3D systolic array, assume that all atomic operations are of the same length, and the time needed for atomic operations is t_3. Then the atomic cycle time for the 3D systolic array is t_3, and the time for the molecular operation (t_{3m}) is:

$$t_{3m} = M t_3$$

if the molecular operation is decomposed into M atomic operations. The total time (T_3) for computation (of matrix **C**) on a 3D systolic array will be:

$$T_3 = (3N + M - 3)t_3$$

Note, the time for computation of matrix **C** on a 3D systolic array will be the same as the time for computation of matrix **C** on a 2D systolic array, if M is equal to one. The plots for

$$\frac{T_2}{T_3} = f(N; M, t_2, t_3)$$

are shown in Fig. 8 for different values of the ratio t_2/t_3 and different values of the parameter M. From these plots one can see that the latency of the 3D systolic array is smaller than that of the 2D systolic array. One can also see from these plots that the ratio T_2/T_3 increases asymptotically with the problem size (N), and the value of asymptote increases with M. The difference in the latency of a 2D and a 3D systolic array increases rapidly with the problem size, and thus 3D systolic array for matrix multiplication proves to be faster than 2D systolic array for matrix multiplication.

5.2 Comparison of area

Now we develop equations for area[1]. First, assume that the area of a cell in the 2D systolic array is given by a_2, and a_3 is the area of a cell in any plane of the 3D systolic array. Now the total area of the 2D systolic array of Fig. 6a (A_{2a}), the total area of the 2D systolic array of Fig. 6b (A_{2b}), and the total area of the 3D systolic array (A_3) are given by the following equations:

$$A_{2a} = (3N^2 - 3N + 1)a_2$$

$$A_{2b} = N^2 a_2$$

$$A_3 = N^2 M a_3$$

The plots for

$$\frac{A_{2a}}{A_3} = f(N; M, a_2, a_3)$$

and

$$\frac{A_{2b}}{A_3} = f(N; M, a_2, a_3)$$

are shown in Figs. 9 and 10, respectively, for different values of the ratio a_2/a_3, and different values of the parameter M. From these plots one can see that the area of the 3D systolic array is less than the area of the 2D systolic array of Fig. 6a, but the area of the 3D systolic array is greater than that of the 2D systolic array of Fig. 6b. It is interesting to note that the ratio A_{2a}/A_3 increases asymptotically with N, but the ratio A_{2b}/A_3 is constant for different values of N. Another interesting point to note is that the ratios A_{2a}/A_3 and A_{2b}/A_3 increase with decreasing M. On the other hand, the ratio

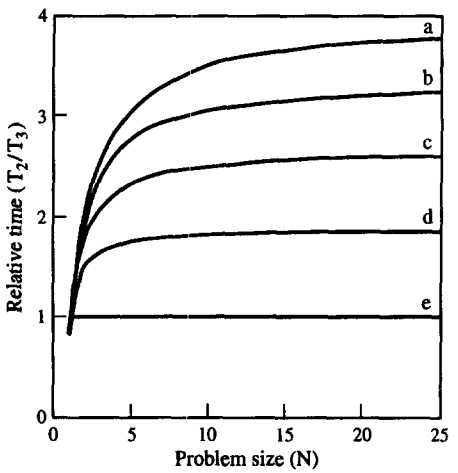

Fig. 8. Plots of T_2/T_3 vs. N. The values of the different parameters are: (a) $t_2 = 4t_3$ and $M = 5$; (b) $t_2 = 3.4t_3$ and $M = 4$; (c) $t_2 = 2.7t_3$ and $M = 3$; (d) $t_2 = 1.9t_3$ and $M = 2$; (e) $t_2 = t_3$ and $M = 1$. Note, $t_2 = M(1 - x)t_3$, where x is the overhead associated with using a 3D systolic array.

[1] The reader is referred to Section 4.2 for our interpretation of the area comparison for 2D and 3D systolic arrays.

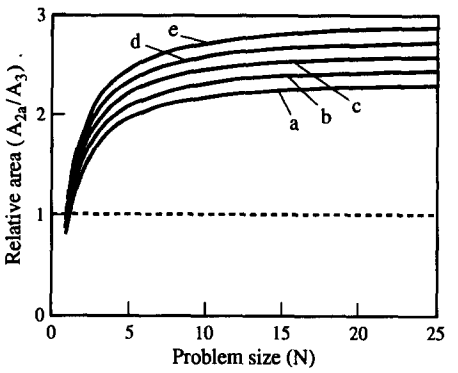

Fig. 9. Plots of A_{2a}/A_3 vs. N. $A_{2a}/A_3 = 1$ is shown by the dotted line. The values of the different parameters are: (a) $a_2 = 4a_3$ and $M = 5$; (b) $a_2 = 3.4a_3$ and $M = 4$; (c) $a_2 = 2.7a_3$ and $M = 3$; (d) $a_2 = 1.9a_3$ and $M = 2$; (e) $a_2 = a_3$ and $M = 1$.

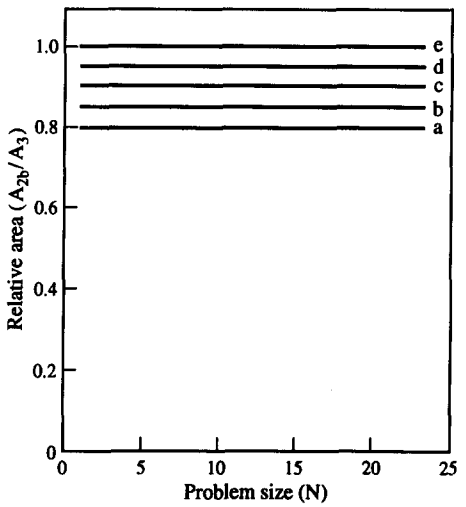

Fig. 10. Plots of A_{2b}/A_3 vs. N. The values of the different parameters are: (a) $a_2 = 4a_3$ and $M = 5$; (b) $a_2 = 3.4a_3$ and $M = 4$; (c) $a_2 = 2.7a_3$ and $M = 3$; (d) $a_2 = 1.9a_3$ and $M = 2$; (e) $a_2 = 1a_3$ and $M = 1$.

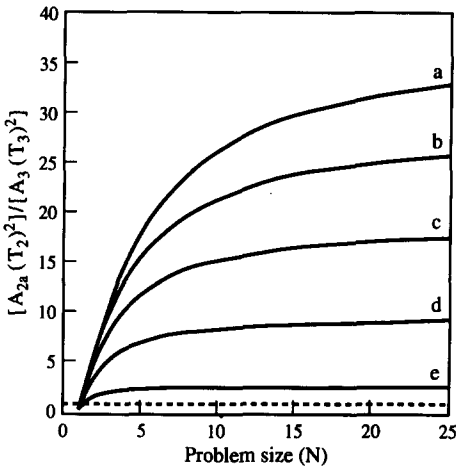

Fig. 11. Plots of $A_{2a}T_2^2/A_3T_3^2$ vs. N. $A_{2a}T_2^2/A_3T_3^2 = 1$ is shown by the dotted line. The values of the different parameters are: (a) $a_2 = 4a_3$, $t_2 = 4t_3$ and $M = 5$; (b) $a_2 = 3.4a_3$, $t_2 = 3.4t_3$ and $M = 4$; (c) $a_2 = 2.7a_3$, $t_2 = 2.7t_3$ and $M = 3$; (d) $a_2 = 1.9a_3$, $t_2 = 1.9t_3$ and $M = 2$; (e) $a_2 = a_3$, $t_2 = t_3$ and $M = 1$.

T_2/T_3 increases with increasing M. Thus, one can conclude that there is a clear tradeoff involved between time and area, and therefore it is important to look at the overall area–time performance of 2D and 3D systolic arrays.

5.3 Area–time tradeoff

As we mentioned earlier, there is always a tradeoff involved between time and area. This tradeoff can be studied from the plots of $A_{2a}T_2^2/A_3T_3^2$ vs. N, and $A_{2b}T_2^2/A_3T_3^2$ vs. N, where

$$\frac{A_{2a}T_2^2}{A_3T_3^2} = f(N; M, a_2, a_3, t_2, t_3)$$

and

$$\frac{A_{2b}T_2^2}{A_3T_3^2} = f(N; M, a_2, a_3, t_2, t_3)$$

These plots are shown in Figs. 11 and 12, respectively, for different values of the ratio t_2/t_3, for different values of the ratio a_2/a_3, and different values of the parameter M. From these plots one can see that the overall performance of a 3D systolic array increases asymptotically with N and the value of the asymptote increases with M, for M greater than one. Also, as we

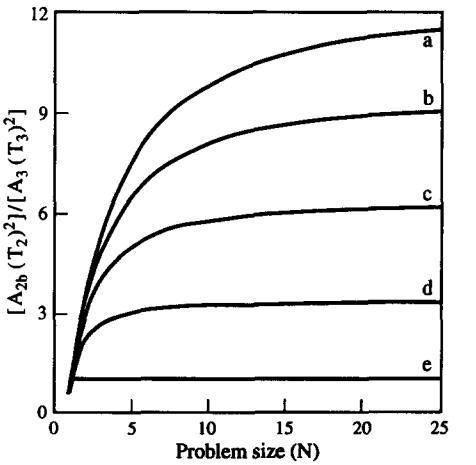

Fig. 12. Plots of $A_{2b}T_2^2/A_3T_3^2$ vs. N. The values of the different parameters are: (a) $a_2 = 4a_3$, $t_2 = 4t_3$ and $M = 5$; (b) $a_2 = 3.4a_3$, $t_2 = 3.4t_3$ and $M = 4$; (c) $a_2 = 2.7a_3$, $t_2 = 2.7t_3$ and $M = 3$; (d) $a_2 = 1.9a_3$, $t_2 = 1.9t_3$ and $M = 2$; (e) $a_2 = a_3$, $t_2 = t_3$ and $M = 1$.

mentioned earlier, the performance of the 2D systolic array of Fig. 6b is the same as the performance of the 3D systolic array for M equal to one (Fig. 12e).

6. Conclusion

Systolic array architectures were initially introduced to solve the latency problems in special purpose processors. Since systolic arrays exploit concurrency in the problems, they are faster, especially for solving the compute-bound problems. Furthermore, as the systolic arrays are regular, they are relatively simpler to design and relatively cheaper to implement in terms of cost per processing element. However, most work so far has been done with planar systolic arrays. There are some inherent limitations to the speed, extensibility and partitionability of planar systolic arrays. To solve these problems, researchers have recently introduced the concept of 3D systolic arrays.

In this paper we have shown a 3D systolic array implementation of 2D matrix multiplication. We have also shown that the overall performance (in terms of time and area) of the 3D systolic array for 2D matrix multiplication is (relatively speaking) better than that of the 2D systolic array.

After studying the results presented in this paper, we believe that other problems can also benefit from 3D systolic arrays. We also believe that progress in 3D technologies (3D VLSI and 3D packaging of 2D VLSI) will make the 3D systolic arrays more competitive in terms of cost.

To understand fully the advantages of 3D systolic arrays, further research work is needed in the area of special purpose architectures and 3D VLSI. One area that needs to be investigated further is the development of systematic methodology to transform an algorithm into a 3D systolic array. A lot of good work was done in developing systematic methodologies to transform an algorithm into a 2D systolic array [20, 21]. On a similar basis, we need to develop a methodology to transform an algorithm into a 3D systolic array. Some work has been done in this area (see [13]) but further research is required. A second area that needs to be investigated further is the development of ways to use 3D systolic arrays in the form of a macropipeline, to solve complex problems. In some problems it is important to find ways to combine simple systolic arrays in the form of a macropipeline, to solve complex problems. A third area that needs to be investigated is fault tolerance capabilities of 3D systolic arrays. It is also important to find ways to design a standard fault tolerant programmable cell that can be used in different 3D systolic arrays.

References

[1] H.T. Kung and C.E. Leiserson, Systolic arrays (for VLSI), *Sparse Matrix Proc. 1978, Society for Industrial and Applied Mathematics,* 1979, pp. 256–282.

[2] H.T. Kung, Why systolic architecture?, *IEEE Computer,* 15(1) (1982) 37–46.

[3] A.J. Blodgett and D.R. Barbour, Thermal conduction module: A high-performance multilayer ceramic package, *IBM J. Res. Dev.,* 26(1) (1982) 30–36.

[4] J.F. Gibbons, SOI — A candidate for VLSI? *VLSI Design,* 3 (1982) 54–55.

[5] G.R. Nudd and R.D. Etchells, Three-dimensional VLSI architecture for image understanding, *J. Parallel Distributed Comput.,* 2 (1985) 1–29.

[6] M. Aboelaze and B.W. Wah, Complexities of layout in three-dimensional VLSI circuits, *Proc. Int. Symp. Circuits Systems,* Philadelphia, PA, May 1987, pp. 543–546.

[7] T. Kaczorek, *Synthesis of Multivariable Systems and Multidimensional Systems,* John Wiley, New York, 1994.

[8] F.T. Leighton and A.L. Rosenberg, Automatic generation of three-dimensional circuits layout, *IEEE Int. Conf. on Computer Design: VLSI in Computers,* 1983, pp. 633–636.

[9] F.T. Leighton and A.L. Rosenberg, Three-dimensional circuits layout, *SIAM J. Comput.,* 15 (1986) 793–813.

[10] F.P. Preparata, Optimal three-dimensional VLSI layout, *Math. System Theory,* 16 (1983) 1–8.

[11] A.L. Rosenberg, Three-dimensional integrated circuits, in H.T. Kung, R.F. Sproull and G.L. Steele,

Jr. (eds.), *VLSI Systems and Computations,* Computer Science Press, Rockville, MD, 1981, pp. 69–80.

[12] A.L. Rosenberg, Three-dimensional VLSI: a case study, *J. ACM,* 30(3) (1983) 397–416.

[13] Y.X. Wang, Using three dimensional systolic arrays, *Purdue University Internal Report,* January 1986.

[14] A. Kokubu, Japanese three-dimensional device project, *Preconference Tutorial of the 13th Int. Symp. on Computer Architecture,* Tokyo, Japan, June 1986.

[15] J.A.B. Fortes and C.S. Raghavendra, Gracefully degradable processor arrays, *IEEE Trans. Comput.,* C-34(11) (1985) 1033–1044.

[16] S. Lakhani, 3D systolic arrays, *Purdue University Master Thesis,* 1987.

[17] J.D. Ullman, *Computational Aspect of VLSI,* Chapter 2, Computer Science Press, Rockville, MD, pp. 42–79.

[18] G.J. Li and B.W. Wah, The design of optimal systolic array, *Trans. Comput.,* C-34(10) (1985) 66–75.

[19] S. Y. Kung, On supercomputing with systolic wavefront array processor, *Proc. IEEE,* 72(7) (1984) 867–884.

[20] H.T. Kung and R.L. Picard, Hardware pipelines for multi-dimensional convolution and resampling, *Proc. IEEE Computer Society Workshop on Computer Architecture for Pattern Analysis and Image Database Management,* Nov. 1981, pp. 273–278.

[21] J.A.B. Fortes, K.S. Fu and B.W. Wah, Systematic design approaches for algorithmically specified systolic arrays, *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing,* March 1985, pp. 300–303.